Developing a Chemical Reconnaissance Behavior for Unmanned Ground Vehicles Using the OneSAF Battlefield Simulation Tool

Dr. MaryAnne Fields
Mr. Tom Haug

ABSTRACT

One of the goals of the U.S. Army Ground Robotics Research Program is to develop individual and group behaviors that allow the robots to contribute to battlefield missions such as reconnaissance. By using simulation tools, we are able to develop, debug and test behaviors before porting them to actual robotic platforms. This affords the researchers the opportunity to expeditiously evaluate the behaviors in a diverse set of environments and to explore variations in behavior algorithms without tying up limited robotic resources or putting these robotic vehicles at risk. This report describes our efforts to develop a proof-of-principal behavior initiated in simulation then ported to a team of surrogate robotic platforms. The issues discovered in porting the algorithm from simulation to the robotic platform are discussed. The report concludes with a discussion of possible extensions to the basic tool.

KEYWORDS: ***Unmanned Ground Vehicle,*** *Tactical Behaviors, Autonomous, One Semi-Automated Forces, Reconnaissance, NBC Mapping, Scout*

1. INTRODUCTION

The primary focus of the U.S. Army Ground Robotics Research Program is autonomous mobility. As unmanned ground vehicles (UGVs) become more capable of autonomously negotiating complex cross country and urban environments, it becomes possible to develop individual and group behaviors that will allow the robots to contribute to battlefield missions such as reconnaissance. Since experimental time on the current robotic vehicles, the experimental unmanned ground vehicles (XUV), is divided between many organizations, it is essential that we develop a simulation tool that will allow us to develop and test behaviors in simulation before porting them to the actual vehicle. Other benefits of developing behaviors in simulation are the ability to expeditiously exercise the behavior in varied environments, and the opportunity to make mistakes without catastrophic effects on the robot. In this report, we describe our efforts to develop a chemical reconnaissance behavior for a team of three XUVs.

Some general background on the chemical reconnaissance mission, as outlined in the Scout Platoon field manual, is presented as the basis for the robotic behavior algorithm. The One Semi-Automated Forces (OneSAF) simulation tool and the modifications that have been made to OneSAF to support the behavior development efforts are discussed. The paper includes a discussion of the basic behavior algorithm as well as enhancements required to add robustness to the behavior in the simulation environment. The next step was to port the behavior algorithm to the robotic platform. Although the ultimate target vehicle for the behavior is the XUV, surrogate robots were used to demonstrate the behavior and to facilitate experimentation and the evaluation of the behavior. The issues discovered in porting the behavior will be discussed. The report concludes with a discussion of potential extensions to the basic behavior development tool.

2. BACKGROUND ON THE MISSION

We use the OneSAF simulation tool to develop autonomous behaviors for ground robotic systems. The scout mission was selected as the application to demonstrate the utility of ground robotics and to showcase the advances in autonomous mobility. There are many aspects of the overall Scout mission including reconnaissance, target acquisition and surveillance (RSTA), classification of terrain features and locating obstacles such as minefields or regions contaminated by nuclear, biological or chemical (NBC) weapons. As a proof-of-concept, we focused our initial behavior development efforts on the mission to locate and mark a region contaminated by an NBC weapon, specifically a persistent chemical contamination. This mission can be accomplished with minimal operator intervention and integrated with the existing Demo III XUV using currently available technologies.

We adapted our behavior from the description of the manned chemical reconnaissance mission given in the FM 17-98 Scout Manual [1]. This mission, as performed by manned scouts, uses existing chemical sensors that alert the user of the existence of contamination. We assume that these sensors can be mounted on the XUV and that signals from these sensors can be interpreted by the XUV software. In this work, we focus on locating and mapping a persistent contaminating agent that has already been released and settled on the surface of the terrain. The tactical use of these agents is similar to the use of minefields. Such agents are used to canalize friendly forces; or to deny them access to intersections, likely avenues of approach, or other key terrain features. Figure 1 shows graphically how the mission is accomplished.

The mapping portion of the mission is accomplished with three vehicles. The first, designated as the base vehicle, establishes the near and far side limits and the base line. The wingmen determine the left and right limits. The movement of the three vehicles is conditional, based on several factors including positive or negative detections; the vehicle's direction of travel; the reconnaissance direction of travel; the base line; and the far side limit. The left, right, and near side limits are defined by the bounding of the vehicles but are not considered in the decision process. The end result is a rectangle, formed by the four limit lines, that bounds the contamination. This mission is only intended to provide sufficient information to maneuver the main body, not to provide a detailed map of the contamination. The step-by-step approach and the hazardous nature of the task both make this a desirable mission for a robotic vehicle.

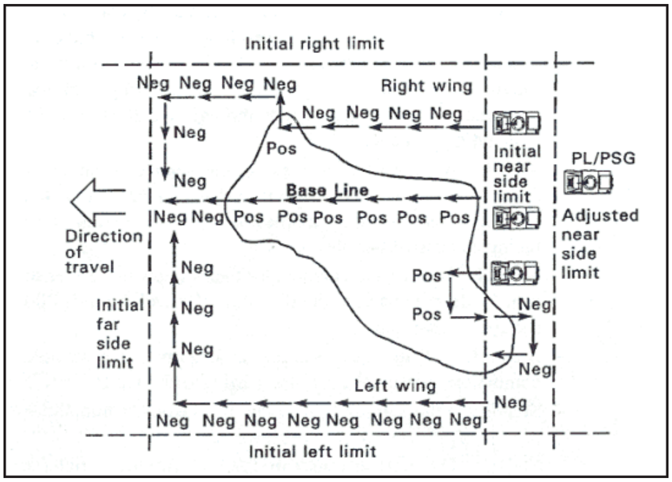


Figure 1. Example of Mapping of Chemical Contamination [1]

3. THE ONESAF SIMULATION TOOL

3.1 BASELINE FEATURES

OneSAF [2] is an interactive battlefield simulation tool developed by the Simulation Training and Instrumentation Command (STRICOM) that simulates the behavior of units, their vehicles, and their weapon systems to a level of realism sufficient for training and combat development. It provides users with the capability to create and control units ranging in size from individual combatants and platforms through battalions. The simulation package also includes a representation of the physical environment, including terrain, diurnal cycle and weather, and its effect on simulated activities and behaviors.

OneSAF has many desirable features for developing and testing robotic behaviors. It is an easy-to-use, interactive tool that allows users to design test scenarios. Currently, there are several hundred different types of units that can be used in these scenarios. These units range in size from individual soldiers to battalions. The units include both air and ground systems and represent both US and foreign systems. The actions of these units can be controlled by the user or, to a limited extent, controlled by OneSAF behavior algorithms. Users can add new units and behavior algorithms to the base systems to support specific projects. There are many terrain databases available for OneSAF. These terrain databases include U.S. Army installations such as Ft. Knox, Ft. Hood and the National Training Center as well as parts of Europe and Asia. In addition, commercial packages such as MultiGen Creator[tm 1][11] can be used to provide 3-dimensional visualization of the terrain databases.

Unfortunately, OneSAF does have limitations as a tool for developing and testing robotic behaviors. These limitations can be grouped into two categories – terrain database limitations and entity behavior limitations.

Most of the terrain databases that are available for OneSAF have elevation posts spaced 30-125 meters apart. This results in very a "smooth" terrain surface that does not accurately model the terrain encountered by a small vehicle. Many terrain features, such as trees, wooded areas, roads, rivers, and buildings are "layered" on top of the elevation grid as linear or polygonal abstract features. Although these abstract features do affect the activities of the simulated entities, they are not directly sensed by the sensory equipment attached to entities. It is difficult to examine the robustness of behaviors that involve autonomous mobility without including a model of how the driving sensors acquire information about the environment. Also, most OneSAF terrain databases do not contain ditches, holes, rocks, boulders, and other small obstructions that present significant obstacles for ground robots.

The current OneSAF mobility behavior algorithms assume a competent human driver is controlling the system. This driver model "perceives" and responds appropriately to obstacles in the terrain, updating the vehicle position and velocity several times a second. In fact, since the driver is assumed to be competent, most OneSAF terrain databases do not contain small mobility obstacles to stimulate the driving algorithms. We cannot assume a competent driver for the Demo III program since a major issue is the robustness of its driving algorithms. We have not fully investigated other behavior algorithms in OneSAF, however, many of the algorithms are trying to simulate *human* actions so they may use information and intelligence not yet available to ground robots. In general, we would like to replace the OneSAF behavior algorithms with a better representation of robotic behavior.

3.2 ARL Extensions

We have extended the basic features of the OneSAF simulation code to better represent ground robotic features. Our work can be divided into two categories – terrain modifications and robot-specific modifications. The terrain modifications overcome some of the limitations of the terrain databases described in the previous section, providing the

[1] MultiGen Creator is a trademark of the MultiGen-Paradigm Corporation, 2044 Concourse Drive San Jose, CA 95131.

simulated robot with a rich environment containing both large and small obstructions that need to be sensed and incorporated into its mobility plan. There are many different approaches to modifying the OneSAF terrain databases to support mobility analysis for robotic vehicles, a detailed discussion of these modifications is found in Fields [7]. In this section, we briefly describe the mobility obstacle editor that we used in analyzing and developing this behavior algorithm.

The mobility obstacle editor allows researchers to introduce two types of obstacles, positive obstacles, representing rocks, bushes, and other obstacles above the ground plane, and negative obstacles, representing ditches, culverts, and other holes in the ground plane to an existing terrain database to stimulate the perception and planning processes on the robotic vehicle. . Using the editor shown in Figure 2, researchers can control the size, shape, number, and distribution of these obstacles. Positive obstacles are show as dark gray, negative obstacles are shown in light gray. By setting the *detectability* and *average detection distance* parameters in the editor window, the researchers control the detection of the obstacles. Using the obstacle editor several times results in the heterogeneous group of obstacles such as the distribution shown in Figure 2. All the information for the obstacles is saved so that the distribution can be duplicated in subsequent simulations.

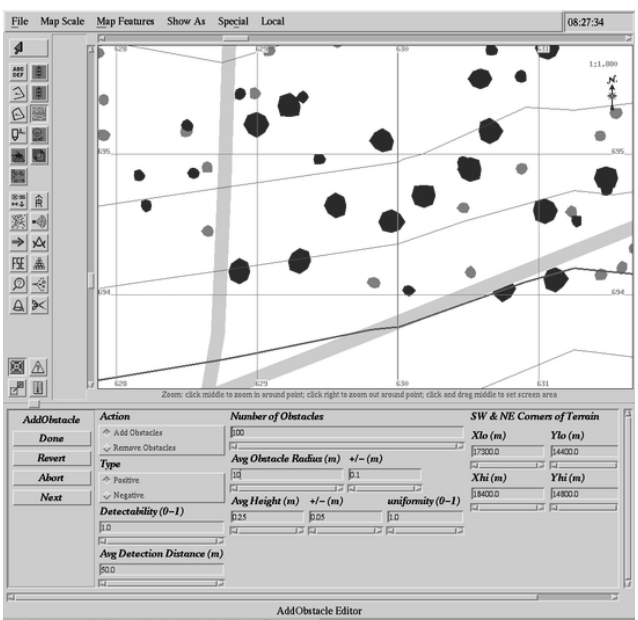


Figure 2. The Obstacle Editor.

In this research, we needed a method to contaminate a region on the simulated battlefield. By designing an editor similar to the obstacle editor, we can place contaminated regions on the battlefield. A contamination is shown in Figure 4. The parameters shown in the editor window determine the size, shape, and location of the region. On the map, the contaminated area is indicated by the gray polygon. Again, the parameter settings can be saved for use in other simulations and the editor can be used multiple times. In this research, we use both the obstacle and contaminate editors to test and debug the chemical reconnaissance behavior.

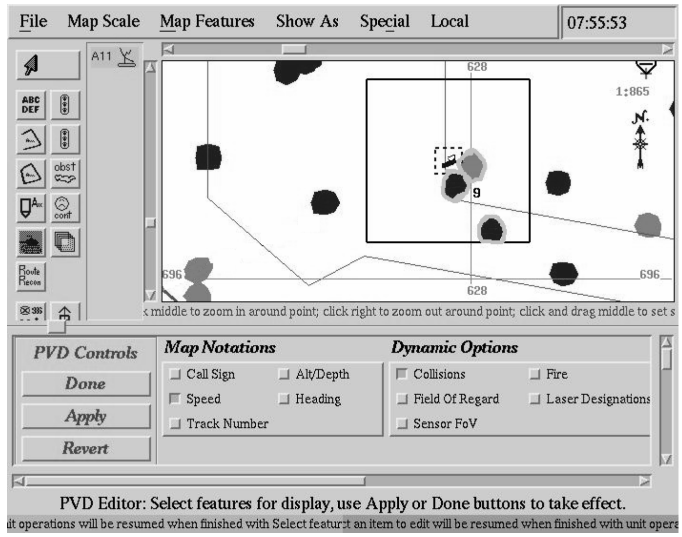


Figure 3. A OneSAF Simulation Display Showing the World Model Information for an Unmanned Ground Vehicle

In addition to the obstacle and contaminate editors, we developed algorithms of robotic driving perception and robotic mobility which have been documented previously [8]. These algorithms model the perception and planning processes of the robot. The perception algorithms are “aware” of the mobility obstacles previously discussed. At each time step, the robot constructs a world model showing detected obstacles and features within a 50 m radius of the robot. The detection of a specific obstacle is a random variable whose probability distribution function is specified by the detectability parameters. Figure 3 shows a world model superimposed on the terrain map. Polygons outlined in light gray have been detected; the remaining polygons are out of the range of the driving sensor.

4. ALGORITHM DESCRIPTION

In this section, we describe the basic algorithm for a OneSAF robotic team to locate and map a contaminated region on the ground. There are some differences between the robotic mission and the manned scout mission. Our simulation begins with the three-vehicle section that is normally organized after initial detection. This eliminates the need to model the entire platoon and simplifies the process of reorganizing the platoon once chemical contamination is detected. Later, we can extend our basic behavior to include a platoon of vehicles participating in the mission. We also did not require the robots to use a bounding overwatch movement technique. Bounding overwatch can be added later without changing the underlying mapping behavior.

We have broken the mapping algorithm into five distinct phases: (1) Locating the Area, (2) Regrouping, (3) Establishing the Baseline, (4) Mapping the Region, and (5) Completing the Mission. Each phase represents a distinct behavior involving one or more of the robots. For now, we assume that transition between the phases is instantaneous – in reality, the transition times depend on the speed and reliability of the robots' communication systems.

4.1 Phase I: Locating the Area

In this phase the robots must find the contaminated region. We assume that the soldier/operator has intelligence information giving an approximate location of the contaminated area. From this information, the operator specifies an initial rally point that forces the robots to cross the suspected area. Figure 4 shows an example map – the suspected area is a large circular region shaded gray, the actual contaminated region is darker irregularly shaped polygon shaded. Keep in mind that the robot and the operator

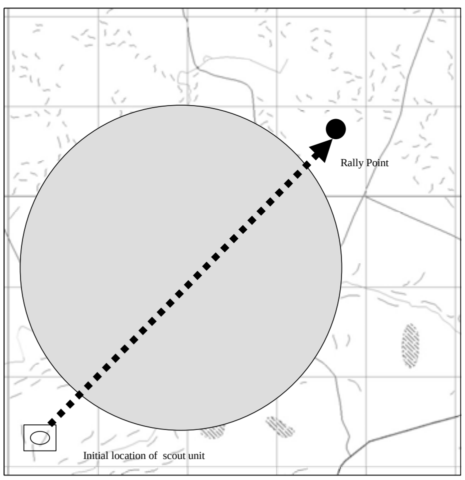


Figure 4. A Battlefield Map Showing Region of Suspected Contamination (gray) and Actual Contamination (green)

don't know the location of the contamination a priori. Just as on the battlefield, it is possible for the operator to select a search path that misses the contamination. In this case the operator picked a rally point that forced the robots to travel through the actual contamination. During this phase, the robots move toward the rally point along parallel paths. The spacing between the robots can be specified by the operator. This phase ends when one of the robots makes contact with the contaminated region or all the robots reach the rally point. The nuclear, biological, chemical (NBC) sensor model assumes perfect instantaneous detection so that any contact with the contaminated region will result in a detection. Unlike the NBC sensor described in Section 2, this sensor samples the environment continuously. If all the robots reach the rally point, it is up to the operator to re-evaluate the mission. He may choose to send the robots through the region again, continue the search to a new rally point or he may choose to abort the mission.

4.2 Phase II: Regrouping

Once the robot team makes contact with the contaminated area, the team reorganizes itself to efficiently map the region. The robot which made the initial contact is designated as the base robot, the other two vehicles are designated as the left wingman and the right wingman. In the current algorithm, the left and right wingman positions as assigned arbitrarily. With future improvements, it will be possible to use information about the relative positions of the robots to make these assignments. In this phase, the base vehicle is stationary. During the second phase, the left and right wingmen prepare for the remainder of the mapping mission by repositioning themselves at a rendezvous point 50 meters behind the current position of the base vehicle. (Note: This differs slightly from the task outlined in the field manual.) Figure 5 shows a robot team positioning themselves in the regrouping phase.

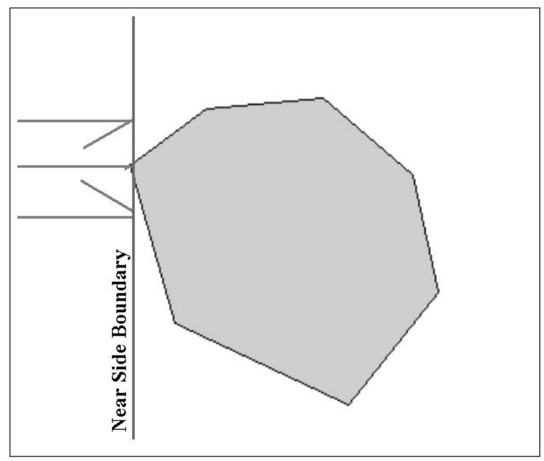


Figure 5. Phase II. The Regrouping Phase

4.3 Phase III: Establishing the Baseline

In the third phase of the mapping mission, the base robot needs to determine the extent of the contaminated region. It travels through the contaminated region towards the rally point to determine the extent of the contaminated region. As a guide to the two wingman, the base robot determines a far side limit 200m beyond the last positive detection, which is perpendicular to the baseline at its current position. Figure 6 shows a robot team at the completion of the third phase and the *Far Side Boundary* line.

4.4 Phase IV: Mapping the Region

The left and right wingmen map the contaminated region in the fourth phase of the mission. Before they start, the wingmen know, from communications with the base robot, the location and direction of the baseline and the location of the far side limit. The left wingman will map the region in a clockwise direction and the right wingman will map the region in a counter-clockwise direction.

To map the contaminated region, the robots execute a series of FIND and REPOSITION steps. In the FIND step, the goal is to find the contamination. Normally in this step, the robot drives in a direction parallel to the baseline until it finds contamination. In the REPOSITION step, the goal is to move 200m from the contaminated region then set up for the next FIND step.

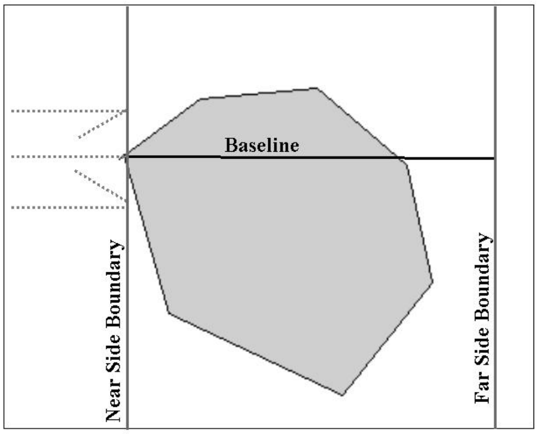


Figure 6. Phase III. Establishing the Baseline and Far Side Boundary

At each positive detection the algorithm must determine the appropriate action. The REPOSITION transition results in a 90^0 turn in a direction that is dependent on the vehicle, left or right wingman. Transitions from FIND-to-REPOSITION and REPOSITION to FIND is all that is required to map a convex shape as shown in Figure 7.

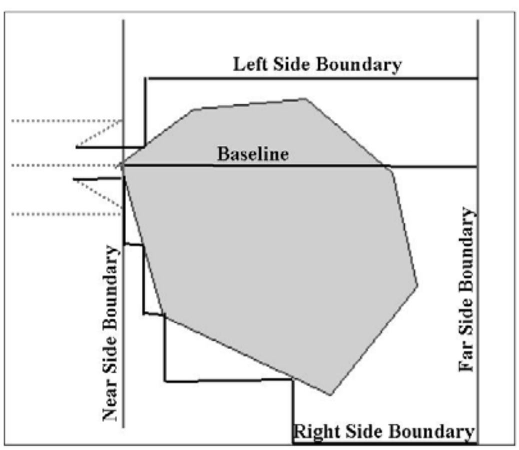


Figure 7. Phase IV. Establishing the Left and Right Side Boundary

In reality, the shape of the contaminated region depends on the prevailing winds, the shape of the terrain, the density and type of features on the terrain surface, as well as the delivery system for the contaminant. A realistic contaminated area might contain concavities, holes, or even disconnected sub-regions. No algorithm can be designed to cover every contingency, but defining few additional transitions greatly enhances the performance of the algorithm.

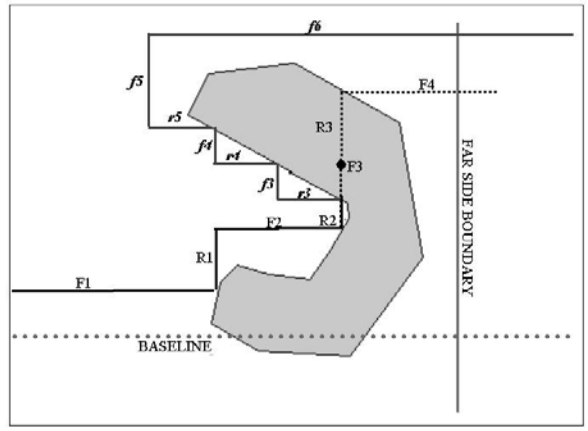


Figure 8. Mapping a Concave Polygon

In Figure 8, two paths are shown for the left wingman. The path designated by F1-4 and R1-3 is the result of the simple approach outlined above. The robot must travel 200m in a REPOSITION mode. This puts the robot in the middle of the contamination at the end of R2. At this point it executes a FIND, F3, turning parallel to the baseline and immediate detection results in a change of state to R3 which gets the robot out of the contamination, but failing to map a large portion of the contamination. The path outlined by F1, R1, F2, R2 and then r3-5 and f3-6 results in mapping the entire area and can be accomplished with only a few added decisions. Adding two additional transitions, FIND-to-FIND, and REPOSITION-to-REPOSITION, allows mapping of an arbitrary polygon. In the new path the robot transitions from R2 to r3 immediately on a positive detection. The FIND-to-FIND transition is required as shown for f5. Continuing along f5 would never accomplish the mission, thus the distance and the direction traveled can be used to trigger a FIND-to-FIND transition.

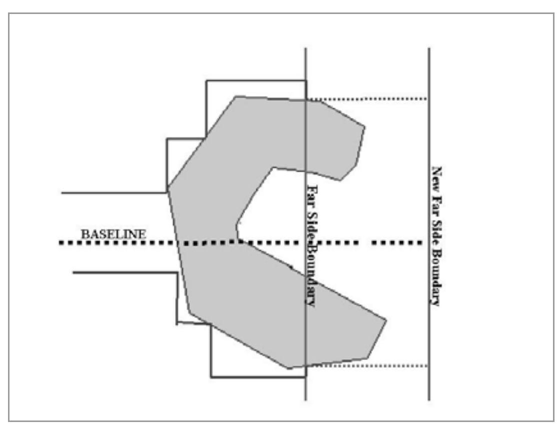


Figure 9. Adjusting Far Side Boundary

4.5 Phase V: Completing the Mission

In the final phase of the mission, the wingmen rejoin the base robot. Figure 9 illustrates another situation that can arise. In this case, the concavity is on the far side and the base robot incorrectly identifies the far side boundary. As the wingmen attempt to rendezvous with the base robot, they contact the contaminated region. In the extended algorithm, this information is passed to the base robot. The base robot travels along the baseline until it can establish a new far side boundary. The wingmen resume the mapping phase, using the new far side boundary as an exit criteria. Presumably, once the robot team has reassembled, they would perform decontamination procedures and prepare for any further missions from the operator.

5. Adding Robustness

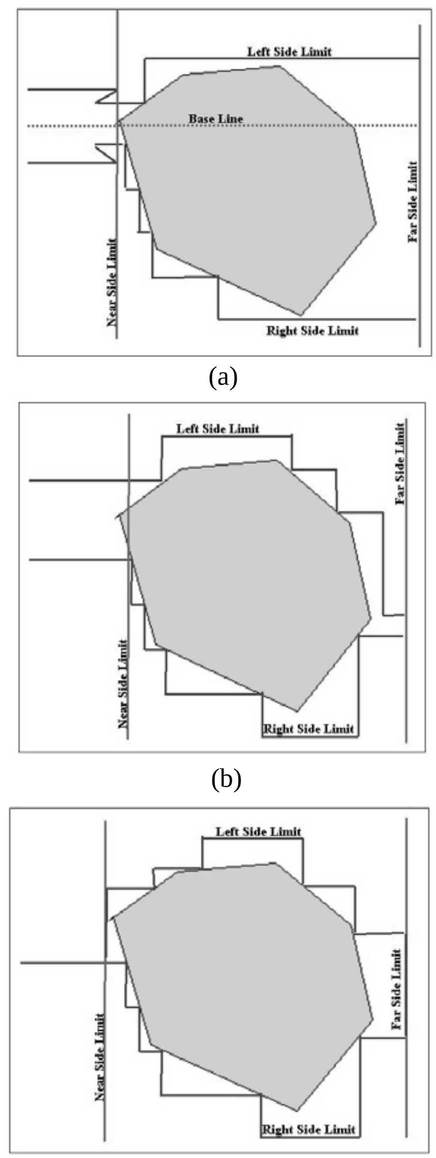
In this section, we describe our efforts to make the mapping algorithm more robust. We concentrate on two problems –the loss of one or more robots, and performing the mission in complex terrain.

5.1 Loss of One or More Robots

The algorithm as presented in the previous sections requires three robots to map a contaminated region. What if some of the robots sustain ballistic damage or break down? Can the mission continue? The mission can continue provided there is a way to monitor the progress and health of the robots.

Extending the current algorithm so that it automatically adapts to the loss of one or more robots requires the mapping behavior to "monitor" the status of the robots, and some adaptation to the mapping phase. Monitoring the status of the robots requires some simple communication between the robots and a central control unit – the robots periodically report their position and status. The robots also report the location of contaminated points, as they encounter these points. If a robot fails to report (or reports that it is damaged), it is assumed to be damaged and unavailable for the remainder of the mission. Adjusting the mapping phase involves fixing the length of the FIND step and using a new exit criteria to end the mapping phase. As an example, consider two cases – the loss of a single robot in the Phase I. and the loss of two robots in Phase I. These two examples can be generalized to cover the loss of robots at any time in the mission.

Consider the loss of a single robot in the Phase I. At this point in the mission, all the robots are looking for the contaminated area- losing a single robot will not significantly change this phase of the mission. However, once the initial contaminated point has been discovered, the robots must reorganize themselves to map the region (Phase II). In the reduced robot team algorithm, we eliminate the role of the base robot and proceed directly to Phase IV, the mapping phase. This leaves the two remaining robots to map the outer edge of the contaminated region without prior knowledge of the location of the far side boundary normally determined by the base robot in Phase III. In the reduced robot team algorithm, the maximum length of the FIND step is a fixed length, forcing the robots to continue probing the area until they are within a fixed distance of each other. Figure 10 shows a comparison between a 3-robot team mapping a contaminated area using the original algorithm and a 2-robot team mapping the region with the adjusted algorithm. In this illustration, the algorithms perform similarly on the left side of the area. On the right side, the reduced robot team algorithm must "map" the area to determine the far side boundary. Also shown in Figure 10 is the procedure for a single robot to perform the mission if two vehicles are lost.



(c)

Figure 10. Mapping the Contaminated Area with: (a) 3-Robot Team, (b) 2-Robot Team, (c) 1-Robot Team

5.3 Performing the Mission in Complex Terrain

In this section, we limit our discussion of complex terrain to terrain surfaces containing a significant number of mobility obstacles that force the robots to deviate from their planned course.

In some ways, performing the mission in complex terrain is similar to performing the mission with a reduced robotic team. Complex terrain can degrade the mobility of one or more of the robots to such an extent that these robots are "lost" to the mission. In the previous discussion, the mapping behavior monitored the status of the three robots as they performed the mission. Adapting the mission to complex terrain requires the behavior to monitor the progress of the robots, as well their status. Progress measures the change in the distance between the robot and its current goal for a given time period.

The goals themselves are a function of the phase of the mission and the robot's particular assignment. In the first phase, progress measures the change in distance between the robots and the rally point. In Phase II only the wingmen are moving, progress for a wingman depends on the distance to the rendezvous point. Recall that the base robot crosses the contaminated region in Phase III moving toward the Phase I rally point. We measure the progress of the base robot in Phase III by measuring the changes in distance between the base robot and the Phase I rally point. In Phase IV, progress depends on the distance between the wingmen and the far side limit. In Phase V, progress depends on the distance between the wingmen and the base robot.

In this work, we have extended the algorithm for two cases – lack of progress of the base robot in Phase III and a significant difference in the progress of the wingmen in Phase IV. In both cases, we will treat lack of progress the same as the loss of a robot – it simplifies the algorithms.

6. Porting the Behavior to a Surrogate Robotic Platform

The next step in this research effort was to port the mapping algorithm from the simulation package to a robotic platform. Two surrogate robots, ATRV-Jr^{TM} from iRobot, Inc., shown in Fig. 12, were selected to demonstrate the mapping behavior on actual robotic platforms. The robots are four wheeled, skid-steered platforms that can be used indoors and outdoors. The ATRV-Jr's sensors include visible spectrum cameras, ultrasonic range sensor array, GPS, an inertial measurement unit, a compass and a tilt sensor. To simplify the hardware and experimental requirements, the cameras were used as surrogate chemical sensors, and yellow plywood disks were used to create contaminated regions. This allowed us to focus on the algorithm and not be distracted by the integration of sensors and the use of chemical simulants.

Since there were only two robotic vehicles available, we demonstrated the reduced robot team algorithm discussed in Section 4.2 (i.e. we assume that the base robot was lost and that the two remaining robots assume the roles of the left and right wingman). Adding the base vehicle would have simplified the mapping process, giving the wing vehicles a baseline and an initial far side limit.

Our experimental setup consisted of the two ATRV-Jr. robots, an operator, and a laptop computer that served as the operator control station. Computer code for the robots and the laptop was written for the Red Hat^{tm} [2] 6.2 operating system and used the Mobility^{tm} [3] C/CORBA libraries provided by iRobot. A wireless Ethernet connected the robots and the operator's control station.

Although the basic chemical reconnaissance behavior is the same for both the real and simulated robots, there are differences in the computer programs controlling each robot. First, even though the intent of the battlefield simulation tool is to model the real world, real and simulated robots interact differently with their respective environments. Service programs such as movement, communication, and sensing need to be written specifically for each environment. The actual behavior program depends on the available programming environment. In OneSAF, behaviors are written as a finite state machine that is translated into C code by the programming environment. On the robot, the chemical reconnaissance behavior was written directly in C.
We wanted to demonstrate that the robot team could conduct the mapping behavior without significant operator involvement. In our experiments, the operator had two roles – send the "start mission" signal to the robots, and act as safety officer for the experiment. Each robot conducted its portion of the mission independently. Communication was minimal. The robots reported contact points to the operator's computer. The contact points were used to draw a map on the OCU screen so that the operator could compare the shape of the mapped region to the shape of the actual region on the ground.

The experiments were successful – the robots were able to consistently map the surrogate contaminated region. We demonstrated both the 2-robot team and the single-robot team mapping procedure. The experiments identified some issues that are important future behavior development work. First is the importance of modeling system latencies. In our behavior, there were two major sources of latencies – latencies associated with detecting the contamination and latencies associated with communication. In the real world, detection is

[2] Red Hat 6.2 is a trademark of the Red Hat Corporation, ,1801 Varsity Drive, Raleigh, NC 27606

[3] Mobility is a trademark of the iRobot Corporation, 32 Fitzgerald Dr., Jaffrey, NH 03452.

not instantaneous –the robot may actually drive into the contamination before it registers a detection. Communication latencies are important to consider for future extensions to this work – monitoring the behavior and adjusting the loss of a robot depend on reliable communication between the robots and the OCU.


Figure 11. ATRV-Jr™ Robots from iRobot

6. Conclusions

This research represents a proof of concept – we were able to develop a behavior using computer simulation then port it to actual robotic platforms. We chose the chemical reconnaissance behavior because it was a potential mission for robot scout unit and because it was algorithmic in nature. We developed the basic algorithm using a modified OneSAF simulation tool. Using simulation experiments to iteratively test our algorithm, we were able to improve the basic algorithm to respond automatically to loss of robots due to attrition or terrain conditions. By using laboratory robots in a controlled environment, we were able to focus on the development of the behavior without having to implement a full autonomous mobility package. The current implementation of the chemical reconnaissance mission on these robots does not take advantage of communications between robots. Future efforts in behavior development will take advantage of the ability of the robots to communicate with each other to more efficiently accomplish tasks.

Developing tactical behaviors in a simulation has many benefits. As discussed in this report, using the enhanced OneSAF simulation to represent current UGV capabilities facilitates the development of behaviors that can be readily transitioned to current platforms. The simulations can also point the way to new technology developments and capabilities required to accomplish more complex behaviors.

This research effort demonstrates that, with a realistic representation of an unmanned ground vehicle and its environment, a computer simulation is a viable tool for building tactical behaviors for unmanned ground vehicles. The current project focused on a single team behavior that had to be designed from scratch using the simulated world to test and debug the algorithm. By structuring our future research so that we develop libraries of common skills and behaviors first, we will be able to combine them into complex individual and group behaviors. As the library of common skills and behavior grows, development and testing time for complex behaviors may decrease since each of the common behaviors and skills will be well characterized.

References .

[1] "FM 17-98 Scout Manual", http://www.adtdl.army.mil/cgi-bin/atdl.dll/fm/17-98/toc.htm, Headquarters, Department of the Army, Washington, DC, April 10, 1999.

[2] Witman, R. and Harrison, C., "OneSAF: A Product Line Appraoch to Simulation", Technical Report, Contract Number DAAB07-01-C-C201. The Miter Corporation, 2001.

[3] T. Ioerger, R. Voltz, and J. Yen, "Modeling Cooperative, Reactive Behaviors on the Battlefield with Intelligent Agents", proceedings of the 9th Conference on Computer Generated Forces and Behavior Representation, 2000.

[4] R.. Hill, J. Gratch,,P. Rosenbloom and R. Whitney, "Flexible Group Behavior: Lessons Learned Building Virtual Commanders", proceedings of the 9th Conference on Computer Generated Forces and Behavior Representation, 2000.

[5] A. Courtemanche amd Charles E. Campbell, "Unified Entity Maneuver", Proceedings of the 9th Conference on Computer Generated Forces and Behavior Representation, 2000.

[6] J. Albus, "4D/RCS Reference Model Architecture for Unmanned Ground Vehicles", Proceedings of the SPIE Vol. 3693, AeroSense Session on Unmanned Ground Vehicle Technology, Orlando FL, April 7-8, 1999.

[7] M. Fields, "Modifying ModSAF terrain Databases to Support the Evaluation of Small Weapons Platforms in Tactical Scenarios", ARL-TR-1996, Army Research Laboratory, Aberdeen, Maryland, August 1999.

[8] M. Fields, "Designing a Behavior Development Environment to Support the Demo III Robotics Program", Proceedings of the SPIE Vol. 4364, AeroSense Session on Unmanned Ground Vehicle Technology, Orlando FL, April 7-8, 2000.

[9] Red Hat, Inc. Red Hat Linux 6.2 Raleigh, NC, 1999.